Resposta Completa / Modelo de Entregáveis da Tarefa 03 – Bootamp DBA

Parte 1, aumentar espaço de tablepace :

system@DB_APP:SQL> *create tablespace TS_APP_NOVO datafile '/opt/oracle/oradata/XE/DB_APP/ts_app_novo_01.dbf' size 100M;*

system@DB_APP:SQL> *select file_name, bytes, autoextensible, maxbytes from dba_data_files where tablespace_name='TS_APP_NOVO';*

```
FILE_NAME                                          BYTES AUT   MAXBYTES
-------------------------------------------------------------- ---------- --- ----------
/opt/oracle/oradata/XE/DB_APP/ts_app_novo_01.dbf               104857600 NO         0
```

==> aumentar o tamanho da tablespace em 10% cfrme solicitado na tarefa, para demonstrar o conhecimento :

system@DB_APP:SQL> *alter database datafile '/opt/oracle/oradata/XE/DB_APP/ts_app_novo_01.dbf'resize 110M;*

```
FILE_NAME                                          BYTES AUT   MAXBYTES
-------------------------------------------------------------- ---------- --- ----------
/opt/oracle/oradata/XE/DB_APP/ts_app_novo_01.dbf               115343360 NO         0
```

Parte 2, solução de monitoramento e registro de esperas por LOCKS - já tenho a tabela TB_PRODUTOS populada (no exercício anterior) , a usarei para simular uma situação de espera por LOCKs :

- uma sessão faz DML num registro que já existe, não há outra acessando portanto a sessão obtém o lock :

PATINHAS@db_app::CONTAINER=DB_APP> *UPDATE APP_OWNER.TB_PRODUTOS SET DESCR_PRODUTO='TV LG' where ID_PRODUTO='P02';*

1 linha atualizada.

- outra sessão tenta fazer DML , ela fica BLOQUEADA :

DONALD@db_app::CONTAINER=DB_APP> *UPDATE APP_OWNER.TB_PRODUTOS SET DESCR_PRODUTO='TV SANYO' where ID_PRODUTO='P02';*

=> a sessão bloqueada fica esperando por um evento chamado "enq: TX - row lock contention" :

system@DB_APP:SQL> *select EVENT, SID, SERIAL#, LOGON_TIME, LAST_CALL_ET, CLIENT_IDENTIFIER, MODULE, PROGRAM, SECONDS_IN_WAIT, STATE, ROW_WAIT_OBJ#, ROW_WAIT_FILE#, ROW_WAIT_BLOCK#, ROW_WAIT_ROW#, BLOCKING_SESSION_STATUS, BLOCKING_INSTANCE, BLOCKING_SESSION, WAIT_CLASS from v$session where event like '%enq%' ;*

```
EVENT                          SID SERIAL# LOGON_TIME          LAST_CALL_ET CLIENT_IDENTIFIER MODULE   PROGRAM     SECONDS_IN_WAIT STATE   ROW_WAIT_OBJ# ROW_WAIT_FILE# ROW_WAIT_BLOCK# ROW_WAIT_ROW# BLOCKING_SE BLOCKING_INSTANCE BLOCKING_SESSION WAIT_CLASS
------------------------------ ---- ------- ------------------- ------------ ----------------- -------- ----------- --------------- ------- ------------ -------------- -------------- ------------- ----------- ------------------ ----------------- ----------
enq: TX - row lock contention  270   41121 27/09/2021 19:12:48         1002                   SQL*Plus sqlplus.exe            1002 WAITING         73526             23             135             1 VALID                       1               152 Application

1 linha selecionada.
```

=> E, também, o próprio LOCK em si fica registrado em outras views / tabelas do sistema, tal como mostradas :

SYSTEM@db_app::CONTAINER=DB_APP> *column sess format a30*
SYSTEM@db_app::CONTAINER=DB_APP> l
```
  1  SELECT inst_id,
  2    DECODE(request,0,'Holder: ','Waiter: ')||sid sess,
  3      id1, id2, lmode, request, type, block, ctime
  4    FROM GV$LOCK
  5   WHERE (inst_id, id1, id2, type) IN
  6    (SELECT inst_id, id1, id2, type FROM GV$LOCK WHERE request>0)
```

```
  7* ORDER BY id1, request;
SYSTEM@db_app::CONTAINER=DB_APP> c/;/
  7* ORDER BY id1, request

SYSTEM@db_app::CONTAINER=DB_APP> l
  1  SELECT inst_id,
  2    DECODE(request,0,'Holder: ','Waiter: ')||sid sess,
  3      id1, id2, lmode, request, type, block, ctime
  4    FROM GV$LOCK
  5   WHERE (inst_id, id1, id2, type) IN
  6    (SELECT inst_id, id1, id2, type FROM GV$LOCK WHERE request>0)
  7* ORDER BY id1, request
SYSTEM@db_app::CONTAINER=DB_APP> /

   INST_ID SESS                         ID1       ID2     LMODE    REQUEST TY      BLOCK     CTIME
---------- ------------------------------ ---------- ---------- ---------- ---------- -- ---------- ----------
         1 Holder: 152                  196616       783         6         0 TX         1      1731
         1 Waiter: 270                  196616       783         0         6 TX         0      1625


2 linhas selecionadas.
```

=> mostrando que esses SIDs/números de sessões indicados acima são os em questão :

```
SYSTEM@db_app::CONTAINER=DB_APP> select sid, serial#, username from v$session where sid in (152, 270);

 SID SERIAL# USERNAME
---- ------- ------------------------
 152     868 PATINHAS
 270   41121 DONALD
```

2 linhas selecionadas.

==>> Apenas simples scripts como os acima demonstrados já Atenderiam à solicitação da tarefa : para atender à solicitação feita na tarefa com notas extras, com uma solução Acima das Expectativas, basta AUTOMATIZAR uma dessas consultas num JOB...

Para isso, o PRIMEIRO conceito necessário para ser entendido é que as ROLEs são DESATIVADAS por default dentro de Stored PL/SQLs , sejam procedures, triggers , functions ou packages, cfrme indicado em https://asktom.oracle.com/Misc/RolesAndProcedures.html : a solução é OU dar os GRANTs necessários para o usuário APP_OWNER que manterá a procedure diretamente sem ser por ROLEs (com a ressalva que o nome das views de indicadores é V_$nomedaview, o V$nomedaview é apenas um sinônimo público), OU criar a procedure com a sintaxe de AUTHID CURRENT_USER , ie, autenticação pelo usuário logado na hora da execução - no exemplo abaixo, usarei a opção de Autenticação direta, criando o stored pl/sql no usuário DB_ADMIN, que em tarefas passadas recebeu as ROLEs que permitem acesso ao dicionário, como SELECT_CATALOG_ROLE :

=> antes de criar tabela, criar tablespace dedicada E dar Quota para o usuário DB_ADMIN :

SYSTEM@db_app::CONTAINER=DB_APP> *create tablespace TS_LOCK_WAITS datafile '/opt/oracle/oradata/XE/DB_APP/ts_lock_waits_01.dbf' size 100M autoextend on maxsize 2G;*

SYSTEM@db_app::CONTAINER=DB_APP> *alter user DB_ADMIN quota unlimited on TS_LOCK_WAITS;*

DB_ADMIN@db_app::CONTAINER=DB_APP> *create TABLE TB_LOCK_WAITS tablespace TS_LOCK_WAITS as (select EVENT, SID, SERIAL#, LOGON_TIME, LAST_CALL_ET, CLIENT_IDENTIFIER, MODULE, PROGRAM, SECONDS_IN_WAIT, STATE, ROW_WAIT_OBJ#, ROW_WAIT_FILE#, ROW_WAIT_BLOCK#, ROW_WAIT_ROW#, BLOCKING_SESSION_STATUS, BLOCKING_INSTANCE, BLOCKING_SESSION, WAIT_CLASS from v$session where 1=2)*
/

=> tabela criada, crio uma procedure que guarda na tabela os waits por locks :

DB_ADMIN@db_app::CONTAINER=DB_APP> *CREATE OR REPLACE PROCEDURE PR_GUARDA_LOCK_WAITS*
authid current_user IS

```
BEGIN
  INSERT INTO TB_LOCK_WAITS (select EVENT, SID, SERIAL#, LOGON_TIME, LAST_CALL_ET,  CLIENT_IDENTIFIER, MODULE, PROGRAM, SECONDS_IN_WAIT,
     STATE,
     ROW_WAIT_OBJ#, ROW_WAIT_FILE#, ROW_WAIT_BLOCK#, ROW_WAIT_ROW#, BLOCKING_SESSION_STATUS, BLOCKING_INSTANCE,
     BLOCKING_SESSION,  WAIT_CLASS from v$session where event like '%enq%'
                    ) ;
  commit;
END;
/
```

==> Agora é só chamar a Procedure a partir de um JOB :

```
DB_ADMIN@db_app::CONTAINER=DB_APP> ed
Gravou file afiedt.buf

  1  BEGIN
  2    DBMS_SCHEDULER.CREATE_JOB (
  3    JOB_NAME => 'JOB_GUARDA_LOCK_WAITS',
  4    JOB_TYPE => 'STORED_PROCEDURE',
  5    JOB_ACTION => 'PR_GUARDA_LOCK_WAITS',
  6    START_DATE => SYSDATE,
  7    REPEAT_INTERVAL => 'FREQ=MINUTELY;INTERVAL=5',
  8    ENABLED => TRUE,
  9    COMMENTS => 'A cada 5 minutos verifica espera por LOCKs.'
 10                      );
 11* END;
DB_ADMIN@db_app::CONTAINER=DB_APP> /
```

=> Consultando imediatamente, eu vejo que o JOB ** foi ** executado já uma vez, por causa do START_DATE indicado :

```
DB_ADMIN@db_app::CONTAINER=DB_APP> select * from TB_LOCK_WAITS;
```

```
EVENT                                                            SID SERIAL# LOGON_TIME          LAST_CALL_ET CLIENT_IDENTIFIER                                 MODULE                                                           PROGRAM
---------------------------------------------------------------- ---- ------- ------------------ ------------ ---------------------------------------------------------- ---------------------------------------------------------------- ------------------------------------------------
enq: TX - row lock contention                                    270   41121 27/09/2021 19:12:48         4686                                                            SQL*Plus                                                         sqlplus.exe

1 linha selecionada.

==> Consultando mais 5 minutos depois , FOI executado novamente :

DB_ADMIN@db_app::CONTAINER=DB_APP> /

EVENT                                                            SID SERIAL# LOGON_TIME          LAST_CALL_ET CLIENT_IDENTIFIER                                 MODULE                                                           PROGRAM
---------------------------------------------------------------- ---- ------- ------------------ ------------ ---------------------------------------------------------- ---------------------------------------------------------------- ------------------------------------------------
enq: TX - row lock contention                                    270   41121 27/09/2021 19:12:48         4686                                                            SQL*Plus                                                         sqlplus.exe
enq: TX - row lock contention                                    270   41121 27/09/2021 19:12:48         4986                                                            SQL*Plus                                                         sqlplus.exe

2 linhas selecionadas.
```

DB_ADMIN@db_app::CONTAINER=DB_APP>

==> E é isso... CLARO que poderiam ser usada a outra query fornecida, Poderia se ter montado uma consula mais profissional, trazendo os DETALHES da sessão bloqueadora lendo novamente a V$SESSION, entre muitas outras Possíveis melhorias, mas o Conhecimento básico é esse que foi demonstrado...